# UBAI PRACTICE - CNN

hs.hwang

2024-11-04

# Table of contents

# 1 CNN

Python , MNIST CNN .

CNN .

, MNIST CNN .

MNIST

MNIST 0 9 .

60,000 10,000 , 28x28 .

MNIST , .

CNN

CNN(Convolutional Neural Network) .

, .

CNN , , , .

### 1.

.

(job) , `filename.sh` . Shell `python_project.sh` .

`.sh` .

```
#!/bin/bash
#SBATCH --nodes=1
#SBATCH --partition=gpu2
#SBATCH --cpus-per-task=56
#SBATCH --gres=gpu:4
#SBATCH --job-name=UBAIJOB
#SBATCH -o ./          /jupyter.%N.%j.out  # STDOUT
#SBATCH -e ./          /jupyter.%N.%j.err  # STDERR
```

```
echo "start at:" `date`
echo "node: $HOSTNAME"
echo "jobid: $SLURM_JOB_ID"

module unload CUDA/11.2.2
module load cuda/11.8.0

python cnn.py 12 256 'relu'
```

. .

STDOUT , STDERR . (directory)
.

.

**#SBATCH --nodes=1**

- , .
- nodes=1 .

**#SBATCH --partition=gpu4**

- Partition .

**#SBATCH --cpus-per-task=14**

- .
- n , 1 CPU/GPU .

**#SBATCH --gres=gpu:1**

- GPU .
- CPU Partition . GPU , .

**#SBATCH --job-name=UBAIJOB**

- .

**echo "start at:" 'date'**

- .

**echo "node: $HOSTNAME"**

- .

`echo "jobid: $SLURM_JOB_ID"`

- jobid .

`module ~`

- Linux .
- GPU , GPU (CPU Partition ) .

`python cnn.py 12 256 'relu'`

- Python .
- .py . cnn.py .
- Python sys sys.argv . sys .
- python {filename}.py .

.py sys , .

- **sys.argv[0]:**
- **sys.argv[n]: (n .)**

cnn.py , 12 256 'relu' sys.argv[1], sys.argv[2], sys.argv[3] .

cnn.py . .

```
import sys
import tensorflow as tf
import keras
import time
import os

from tensorflow.python.keras import layers
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D, MaxPooling2D

start = time.time()

img_rows = 28
img_cols = 28

(x_train, y_train), (x_test, y_test) = keras.datasets.mnist.load_data()
```

```
input_shape = (img_rows, img_cols, 1)
x_train = x_train.reshape(x_train.shape[0], img_rows, img_cols, 1)
x_test = x_test.reshape(x_test.shape[0], img_rows, img_cols, 1)

x_train = x_train.astype('float32') / 255. #
x_test = x_test.astype('float32') / 255.  #

print('x_train shape:', x_train.shape)
print(x_train.shape[0], 'train samples')
print(x_test.shape[0], 'test samples')

batch_size = int(sys.argv[2])
num_classes = 10
epochs = int(sys.argv[1])

y_train = keras.utils.to_categorical(y_train, num_classes) #
y_test = keras.utils.to_categorical(y_test, num_classes) #

model = Sequential()
model.add(Conv2D(32, kernel_size=(5, 5), strides=(1, 1), padding='same', activation='relu', i
model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
model.add(Conv2D(64, (2, 2), activation='relu', padding='same'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten()) # fully connected layer
#                  .
# conv2d           pooling            dense layer          feature map  input
model.add(Dense(1000, activation=sys.argv[3]))  # -> Dense Layer                    .
model.add(Dropout(0.5)) #
model.add(Dense(num_classes, activation='softmax'))
model.summary()

model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
hist = model.fit(x_train, y_train, batch_size=batch_size, epochs=epochs, verbose=1, validatio

score = model.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])

end = time.time() - start

print(end)
```

## 2.

, Python .

, terminal `sbatch` . (job) .

(job) ID .

```
sbatch filename.sh    # ex) sbatch python_project.sh
```

※ *cnn.py* ***pip install tensorflow && pip install numpy*** .

(job) , STDOUT OUT .

OUT , Partition (job) . terminal `squeue` , ID .

n001, n002 … , ( *Resources, Priority* ) .

, .

Partition Partition cpus-per-task, gpu Partition (job) .

STDOUT OUT .